ISAD251 Database Applications Development 2016-17

# Assignment 1 - Oracle and Application Express Application Development

# Placement Application Tracking System (PATSY)

## Scenario

Stage 2 computing students are strongly encouraged to undertake a placement year as part of their studies. In addition to significantly improving the student’s graduate employment prospects, there is strong evidence that undertaking a placement year often leads to a higher degree classification.

There is a need to introduce a system, which will track student activity through the application process from the start of Stage 2 until a placement employment is achieved.

The system must be able to handle students, companies and jobs and track the interaction between all three.

The system should hold the following data about the **student** (mandatory data indicated by \*)

|  |
| --- |
| * \*student record number (SRN) |
| * \*name |
| * **\*programme**[[1]](#footnote-1) |
| * \*date of birth – a student must be at least 18 years of age on 1st June 2016 to be able to undertake a placement |
| * term time address[[2]](#footnote-2) comprising \*house (or flat) number or name and \*postcode |
| * home address comprising house (or flat) number or name and postcode - if different from the term time address |
| * \*mobile telephone number |
| * landline telephone number |
| * \*email address |
| * a note of any placement preferences (e.g. nature of placement, geographical location), |
| * date when a CV was submitted for approval |
| * date when the CV was approved[[3]](#footnote-3) |
| * \*username |

A **company** may have one or more **sites**, e.g. IBM at North Harbour, Hursley Park, Warwick, Feltham, etc. and may advertise placement opportunities at a number of locations. Site data includes its address and postcode.

A **job** vacancy at a company site has a job title, a short job description, a contact for enquiries (either an email address or a telephone number or both), a salary, a start date for the placement (e.g. 1st July 2016), the default time for which is 00:00, an **application method** (CV with covering letter, application form, online application, etc.) and a closing date for applications[[4]](#footnote-4) (e.g. 31st October 2016), the default time for which is 23:59. The start date for the placement and the closing date for applications must be on weekdays (i.e. Monday to Friday).

A student may make an **application** for many jobs and a job may receive many applications. Note that in order to apply for any job, a student must have submitted a CV and have had it approved. Those students with an approved CV are deemed to be *active* in the placement process and those without *inactive*.

For an application for a particular job, an **application history** is required showing the **application status** (from the list below) over time and the date and time of any change in status**.**

application submitted

application withdrawn

applicant invited for interview

applicant invited to assessment centre

applicant rejected

applicant offered position

applicant accepted offer

applicant declined offer

## Tasks

Analyse the above data[[5]](#footnote-5) and implement the resulting tables in Oracle SQL including as much sensible data integrity as possible. You are strongly recommended to consider the use of sequences and triggers.

Design and implement TWO applications in Oracle Application Express 5.0:

**Administrator’s desktop application**

1. to enter and maintain data relating to students, companies, company sites and job vacancies.
2. to browse/search current and past job vacancies.
3. to view a report on the status of a given student and his/her application(s).
4. to provide management reports to show
   * + the name, programme and email address of students without an approved CV, i.e. *inactive* in the placement process
     + the name, programme, email address of students with an approved CV, i.e. *active* in the placement process and the number of applications made
     + the company, site, job title, closing date and number of applications made to date for vacancies with a closing date within the next seven days
     + the name, programme, email address, company, site, job title and start date of students who are placed (i.e. who have accepted an offer)
     + the name, programme and email address of active students who are not yet placed (i.e. who have not accepted an offer)
5. to display a management dashboard with

* a pie chart showing the proportion of placed to unplaced active students
* a stacked bar chart showing the number of active students placed and unplaced by programme
* a calendar of job vacancy closing dates

**Student’s mobile web application** (for use on an iPad Mini for example)[[6]](#footnote-6)

1. to view/browse/search current job vacancies
2. to allow a student to record applications made and maintain his/her application(s) history
3. to allow a student to view a report on the status of his/her application(s)

Your applications should include appropriate validation, interface and navigation features.

You will need to incorporate individual user logins – hence the inclusion of a student username - but you are advised not attempt this until the rest of your application has a high level of functionality.

## Groups

This is a group assignment. Project teams should ideally comprise **four** students, although exceptionally other sizes (usually 4±1). will be permitted by negotiation with the module leader. No allowance for group size will be made in the assessment of the work.

The module leader must be notified of group membership by email to [Ismini.vasileiou@plymouth.ac.uk](mailto:Ismini.vasileiou@plymouth.ac.uk) by 9am on **Monday 10th October 2016**. The module leader may assign any student not in a group by this deadline to a group.

## Deliverables

You should submit **a zip file** (.zip) named using your group name (e.g. APEX99.zip) via the DLE containing THREE files only:

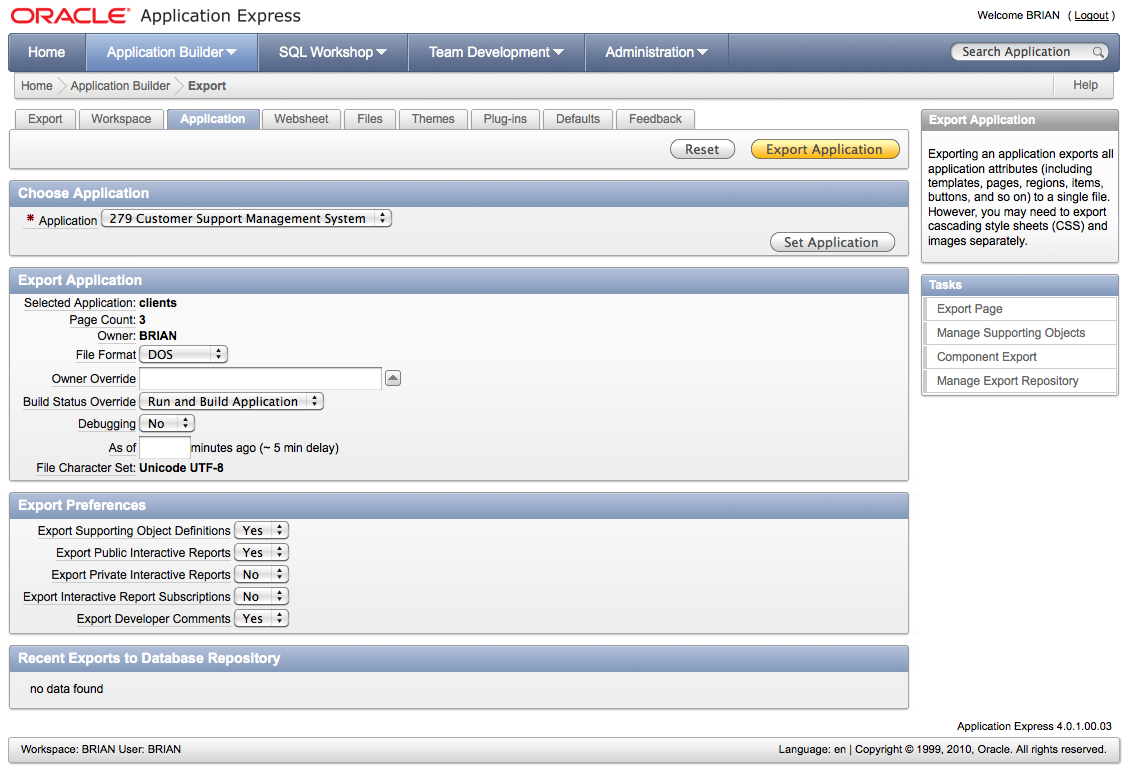
**A single PDF document** (in landscape orientation format[[7]](#footnote-7)) named using your group name (e.g. APEX599.pdf) with (in this order)

1. A list of any additional assumptions you have made which affect your solution.
2. A sensibly sized and legible SQL Developer Data Modeler diagram showing all of your tables and their relationships.
3. Your Oracle CREATE TABLE statementswith any related object creation statements (CREATE SEQUENCE, CREATE TRIGGER)

* You should ensure that your objects are presented in a logical order, i.e. a table referenced by a foreign key (e.g. a lookup table) should appear before the table with the foreign key reference.
* You should pay special attention to the completeness and presentation of these statements as both SQL Developer and APEX Object Browser can generate incomplete DDL output.
* You are strongly advised to refer to the checklist in Appendix B and perform a self-check before submission.
* You must adhere to the presentation guidelines contained in Appendix C. Do not underestimate the time needed to ensure good presentation.

1. Listings of CREATE statements for any other database objects, e.g. views, procedures and/or functions, you may have used.
2. A set of relevant and sensibly sized screen shots showing your application in operation together with a written commentary where appropriate.
3. A critical appraisal of your solution highlighting worthy features, together with any shortcomings and how they might be resolved.

**Export files** (.sql) of each of your applications created using the following settings:



Note that any part of your submission in an incorrect file format may not be marked.

Coursework may be submitted at any time ahead of the deadline time.

Please note the University regulations concerning late submission of coursework.

**Submission deadline: Friday 4th November 15.00 via the DLE**

Note that database objects and applications may need to be accessed in your absence as part of the development and assessment process. It is therefore imperative that your solution is developed on the University’s Oracle server. Solutions developed elsewhere will receive a mark of zero.

Your group will be required to perform a 25-minute defence of your solution at a mutually convenient time (which may or may not be in your normal timetabled slot) **in the week beginning 7th November 2016**. You must ensure that your tables are populated with an adequate amount of sensible test data in advance of this session and it is particularly important that date and time-dependent data is applicable to the date and time of your solution defence. Unless it is unavoidable, all group members should be present at this session.

You will be informed when marking is complete. This will be no later than Friday 9th December 2016 (20 working days from the last solution defence).

## Assessment

The assignment assesses the following Assessed Learning Outcomes for the module, viz. to

1. write effective SQL statements for defining, manipulating and controlling data.
2. design and implement a multi-user database application

The mark for this assignment contributes 50% to the overall mark for the module.

## Marks allocation and assessment criteria

|  |  |
| --- | --- |
| **Deliverable** | **Maximum**  **Marks** |
| **Database diagram and additional assumptions**  To achieve full marks for this section, you must submit a legible SQL Developer database diagram showing all of your tables and their relationships.  Failure to do so will result in a mark of zero for this section. | **5** |
| **Database objects**  To achieve a mark of 40% or above in this section, you must show how you have implemented all of your tables with appropriate table and column names, data types and entity and referential integrity.  To achieve a mark of 60% or above in this section, you must also show how you have implemented a significant amount of data integrity using a variety of methods.  To achieve a mark of 80% or above in this section, you must also show how you have implemented complete data integrity using a variety of methods.  Marks will be reduced for database objects (tables, views, sequences, triggers, procedures and functions) being omitted, poor presentation of code, constraints without explicitly assigned names and deviation from the recommended naming convention. | **30** |
| **Critical appraisal**  To achieve a mark of 40% or above in this section, you must provide a superficial account of the problems areas and worthy features.  To achieve a mark of 60% or above in this section, you must provide a substantial account of the problems areas and worthy features, together with suggestions for improvement.  To achieve a mark of 80% or above in this section, you must provide a complete account of the problems areas and worthy features, together with suggestions for improvement and an indication as to how these may be realised. | **15** |
| **Quality of desktop and mobile web applications including solution defence**  To achieve a mark of 40% or above in this section, you must be able to demonstrate and document that your applications are able to maintain and report on the history of applications.  To achieve a mark of 60% or above in this section, you must also be able to demonstrate and document student, company, site and vacancy maintenance using appropriate interface features and validation and the provision of meaningful management reports and of a management dashboard as specified above.  To achieve a mark of 80% or above in this section, you must also be able to demonstrate and document the use of user roles in your application.  Marks will be reduced for the use of insufficient and inappropriate test data.  Marks will be increased where innovative features have been incorporated. | **50** |
| **TOTAL** | **100** |

Note that use of the Online Peer Assessment System is mandatory for all groups. You must enter some textual feedback as well as scores. Students not contributing to the peer assessment process will be awarded a mark of zero.

## Health Warning

This is an assignment that can not possibly be done at the last minute. It is estimated that it may take up to 200 hours (50 hours per student in a group of four) to complete and you will need to work consistently from the release of the assignment until the deadline.

You must start early even if you do not yet have all the necessary knowledge and skills to complete the assignment.

In the first instance, you should concentrate on implementing working applications – you can add additional features later. You have the opportunity to discuss how your solution can be improved in the critical appraisal.

It is important that you work well as a group especially in terms of task allocation and communication between members. This will ensure that everyone is aware of changes made by others. It may be beneficial to elect a group co-ordinator.

You should note from the module schedule that there are a number of timetabled practical sessions devoted to assignment work during which you will be given feedback on progress so far.

Whilst every effort will be made to ensure that this is not the case, you should assume that some server and network outages will take place, so you must allow for this in your plan of work.

Issued 27th September 2016

## Appendix A

3384 BSc (Hons) Computer & Information Security

3429 BSc (Hons) Computer Science

0746 BSc (Hons) Computer Systems and Networks

2594 BSc (Hons) Computing

4230 BSc (Hons) Computing & Games Development

6007 BSc (Hons) Applied Computing with Professional Experience\*

\* mandatory placement year - others optional but recommended

# Appendix B

## CREATE TABLE checklist

## Meaningful table and column names used

## Appropriate data types used

## Primary key defined

## Foreign key(s) defined if applicable

## Appropriate check constraints defined if applicable

## Appropriate not null constraints defined if applicable

## Appropriate unique constraints defined if applicable

## Column constraints defined where possible and table constraints reserved for multiple column cases

## All constraints named (including NOT NULL constraints)

## Indentation, reserved word identification and non-proportional font used to aid readability

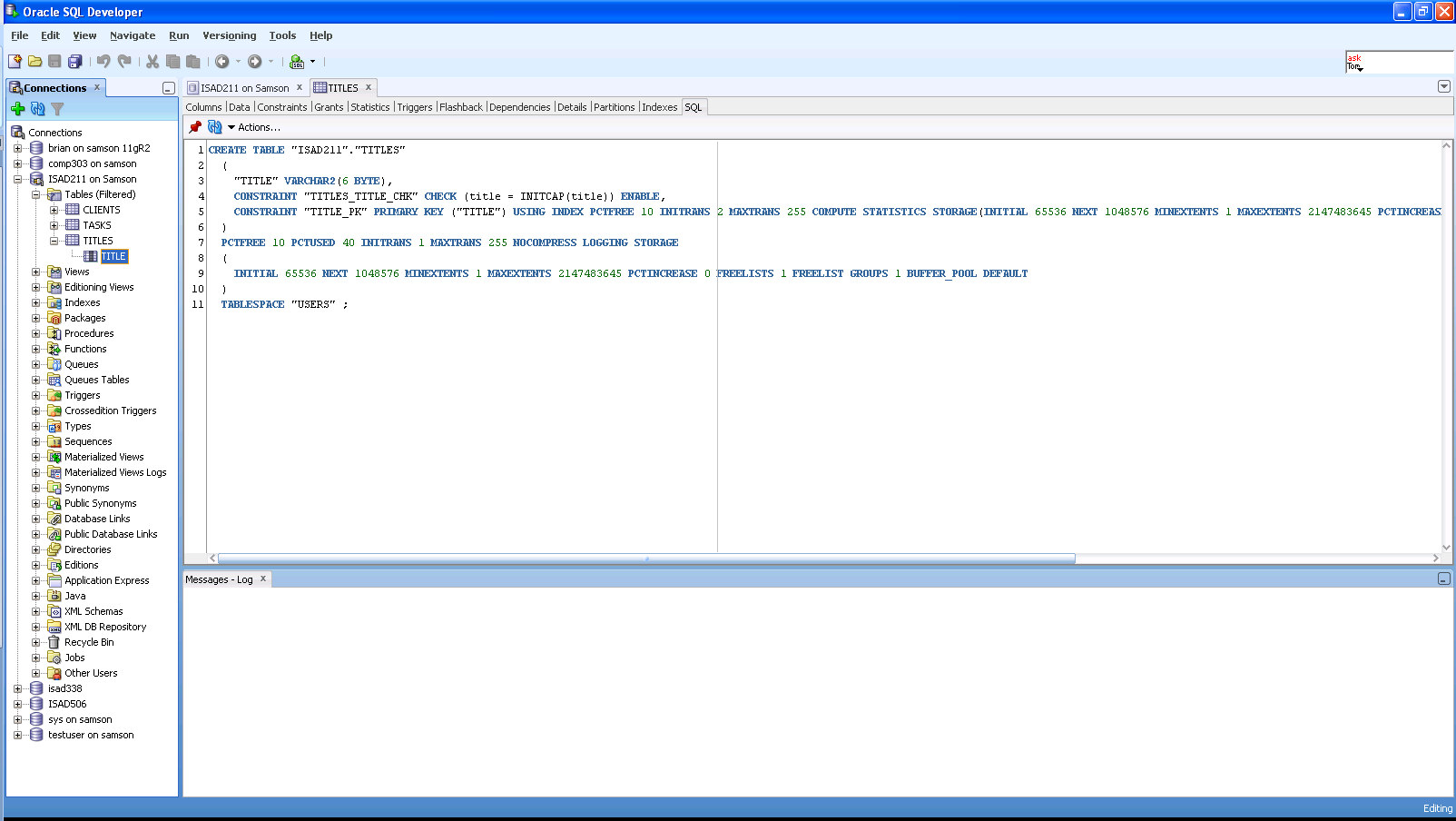
* Consistent use of a naming convention

# Appendix C

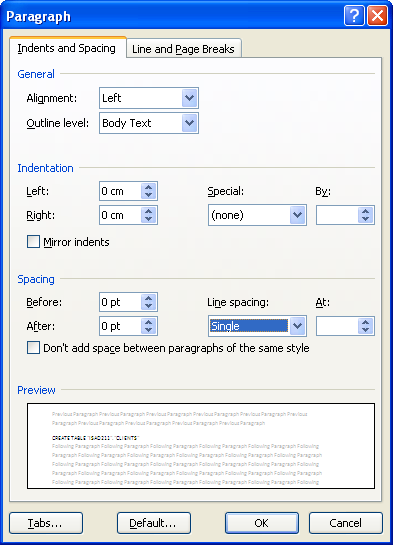
## Presenting SQL CREATE object statements

You must present your statements sensibly and in a logical order. Using the client-titles schema as an example, it is important to present the titles table first because it is referenced by the clients table (Title in clients is a foreign key referencing the primary key in titles).

Use SQL Developer to capture your SQL code for a table by selecting the DDL tab for the table, and copying and pasting the code into Word. You must not include screenshots of your code.



Format the paragraph in Word as follows:



Change to a non-proportional font (e.g. Lucida Console) and delete superfluous content (greyed out below)

**CREATE TABLE "ISAD211"."TITLES"**

**(**

**"TITLE" VARCHAR2(6 BYTE),**

**CONSTRAINT "TITLES\_TITLE\_CHK"**

**CHECK (title = INITCAP(title)) ENABLE,**

**CONSTRAINT "TITLE\_PK" PRIMARY KEY ("TITLE")** USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645 PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 BUFFER\_POOL DEFAULT) TABLESPACE "USERS" ENABLE

**)**

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255 NOCOMPRESS LOGGING STORAGE

(

INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645 PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 BUFFER\_POOL DEFAULT

)

TABLESPACE "USERS" ;

To produce

**CREATE TABLE "ISAD211"."TITLES"**

**(**

**"TITLE" VARCHAR2(6 BYTE),**

**CONSTRAINT "TITLES\_TITLE\_CHK"**

**CHECK (title = INITCAP(title)) ENABLE,**

**CONSTRAINT "TITLE\_PK" PRIMARY KEY ("TITLE")**

**)**

Tidy up formatting and indentation

CREATE TABLE "ISAD211"."CLIENTS"

(

"CLIENT\_ID" NUMBER,

"SURNAME" VARCHAR2(25 BYTE)

CONSTRAINT "CLIENTS\_SURNAME\_NN" NOT NULL ENABLE,

"FORENAME" VARCHAR2(25 BYTE)

CONSTRAINT "CLIENTS\_FORENAME\_NN" NOT NULL ENABLE,

"TITLE" VARCHAR2(6 BYTE),

"DOB" DATE

CONSTRAINT "CLIENTS\_DOB\_NN" NOT NULL ENABLE,

"PHONE\_NO" VARCHAR2(20 BYTE),

CONSTRAINT "CLIENTS\_SURNAME\_CHK"

CHECK (surname = INITCAP(surname)) ENABLE,

CONSTRAINT "CLIENTS\_FORENAME\_CHK"

CHECK (forename = INITCAP(forename)) ENABLE,

CONSTRAINT "CLIENTS\_PHONE\_NO\_CHK"

CHECK (REGEXP\_LIKE(phone\_no, '\(([[:digit:]]{5})\) ([[:digit:]]{6})')) ENABLE,

CONSTRAINT "CLIENTS\_CLIENT\_ID\_PK"

PRIMARY KEY ("CLIENT\_ID") USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645 PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 BUFFER\_POOL DEFAULT) TABLESPACE "USERS" ENABLE,

CONSTRAINT "CLIENTS\_TITLE\_FK"

FOREIGN KEY ("TITLE")

REFERENCES "ISAD211"."TITLES" ("TITLE") ENABLE

)

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255 NOCOMPRESS LOGGING STORAGE

(

INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645 PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 BUFFER\_POOL DEFAULT

)

TABLESPACE "USERS" ;

In this case, as the trigger for the clients table uses a sequence, include the CREATE SEQUENCE statement before the trigger statement. You may not need this if you are using the Oracle 12c identity generation feature.

CREATE SEQUENCE seq\_client\_id

NOCACHE;

CREATE OR REPLACE TRIGGER "ISAD211"."TRG\_CLIENTS" BEFORE

INSERT OR

UPDATE OF dob, phone\_no ON clients FOR EACH ROW

BEGIN

IF INSERTING THEN

IF :NEW.client\_id IS NULL THEN

SELECT seq\_client\_id.nextval

INTO :NEW.client\_id

FROM sys.dual;

END IF;

END IF;

:NEW.forename := INITCAP(:NEW.forename);

:NEW.surname := INITCAP(:NEW.surname);

/\* replace any characters other than digits with an empty string \*/

:NEW.phone\_no := REGEXP\_REPLACE(:NEW.phone\_no, '[^[:digit:]]', '');

/\* adjust to (99999) 999999 format \*/

:new.phone\_no := REGEXP\_REPLACE(:NEW.phone\_no, '([[:digit:]]{5})([[:digit:]]{6})', '(\1) \2');

IF NOT (MONTHS\_BETWEEN(SYSDATE,:NEW.dob) >= 18\*12) THEN

/\* Issue error code (ORA-20000) and message \*/

RAISE\_APPLICATION\_ERROR(-20000, 'Client must be at least 18 years of age');

END IF;

END;

/

ALTER TRIGGER "ISAD211"."TRG\_CLIENTS" ENABLE;

Group a CREATE TABLE statement with its related object creation statement(s) - SEQUENCE, TRIGGER - before moving on to the next CREATE TABLE statement, e.g.

CREATE TABLE statement

CREATE SEQUENCE statement

CREATE TRIGGER statement

CREATE TABLE statement

CREATE SEQUENCE statement

CREATE TRIGGER statement

CREATE TABLE statement

CREATE SEQUENCE statement

CREATE TRIGGER statement

Rather than

CREATE TABLE statement

CREATE TABLE statement

CREATE TABLE statement

CREATE SEQUENCE statement

CREATE SEQUENCE statement

CREATE SEQUENCE statement

CREATE TRIGGER statement

CREATE TRIGGER statement

CREATE TRIGGER statement

1. A programme may have a mandatory placement year. See Appendix A for a set of data. [↑](#footnote-ref-1)
2. You might expect the system ultimately to be able to determine the remainder of any address using a Postcode Address File – you must not implement this. [↑](#footnote-ref-2)
3. Note that you should not store actual CV document itself [↑](#footnote-ref-3)
4. You can assume that once posted on the system, jobs are open to applications. [↑](#footnote-ref-4)
5. Those entities which are in **bold** in above scenario should give you a clue about the required tables. You are encouraged to have your table design verified at an early stage to avoid problems. [↑](#footnote-ref-5)
6. You are advised not to develop your application on a mobile device but to ensure that it operates successfully on a mobile device [↑](#footnote-ref-6)
7. Given that much of your document will look better in landscape (SQL code and database diagram in particular), you should use landscape throughout. This will save you the trouble of adding section breaks. [↑](#footnote-ref-7)